Chapter 9

Application à l'Extraction, la Visualisation, et la Quantification d'Objets Anatomiques Arborescents

Résumé — A partir des outils de visualisation et de mesure du chapitre 7, et des outils d'extraction de surfaces et de sequelettes du chapitre 8, nous étudions plusieurs cas pratiques, dédiés à des objets particuliers. Dans la section 9.2, nous appliquons notre méthode d'extraction de trajectoires et d'arbre du chapitre précédent à l'extraction et la quantification des bronches dans les images médicales 3D de scanners CT. Dans la section 9.3, on considère un tout autre problème: celui du groupement perceptuel, ou la donnée est un ensemble non-structuré de régions de l'image. Nous proposons une méthode de reconstruction de structures arborescentes dans des images médicales tridimensionnelles.

Abstract — Using the several techniques developed in chapters 7 and 8, we develop applications for medical problems. In section 9.2, we apply the complete multiple paths and shape extraction framework of chapter 8 to the segmentation and quantification of airways in 3D multi-slice CT scanner images. Finally, in section 9.3 we consider the problem of *Perceptual Grouping* and contour completion, where the data is an unstructured set of regions in the image. We propose a new method which is illustrated on reconstruction of tree structures in 3D angiography images.

9.1 Application to 3D Vascular Images with Multiscale Vessel Enhancement

In this section, we focus on vascular tree extraction, for accurate determination of vessel width (important in grading vascular pathologies, such as stenosis, or aneurysm). We are particularly interested in using Multiscale Vessel Enhancement techniques of *Frangi et al.* [60].

9.1.1 Medical relevance

All methods developed in this chapter are illustrated on the particular problem of vascular tree extraction in 3D contrast enhanced medical images. The medical interest of this extraction is mostly accurate determination of vessel width. It is an important step in grading vascular pathologies, such as stenosis, or aneurysm.

Stenosis quantification

In the carotid arteries, this quantification determines the choice of stroke treatment. Studies have revealed that patients with severe symptomatic stenosis in the carotids should undergo surgical treatment, and support the relevance of accurate measurement techniques of vascular segments.

Aneurysm quantification

For explanations on this pathology we refer to section 6.1 where they are studied in the case of cerebral vessels. Those pathologies, which are roughly speaking “inflations” of an artery that weak its walls, and can lead to an hemorrhage, occur for example in the brain and, and in the abdominal aorta (see figure 3.12).

Potential: Multiscale Vessel enhancement

For the definition of the speed function for the *Fast-Marching* algorithm, we can use the output of a multi-scale vessel filters based on the Hessian matrix [105, 60]. This paragraph will be illustrated by an application on the dataset shown in figure 9.1[1]. We have used the measure defined by *Frangi et al.* [58] in the following. The symmetric Hessian matrix $\mathbf{H}$ describes local second order intensity variations in the image and is given as:

$$H_{ij} = \frac{\partial^2 I}{\partial x_i \partial x_j} \ , i,j = 1, \ldots, n \tag{9.1}$$

where $I(\mathbf{x})$ is the n-dimensional image. The Hessian matrix defines an ellipsoid where the direction of its smallest axis is the direction of minimal second derivative, that defines the local direction of a tub-like structure. Having extracted the three eigenvalues of the Hessian matrix computed at scale σ, ordered $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$, we define

[1] We would like to acknowledge Dr Wiro Niessen, from Image Sciences Institute, University Hospital Utrecht, Netherlands, who provided this image.

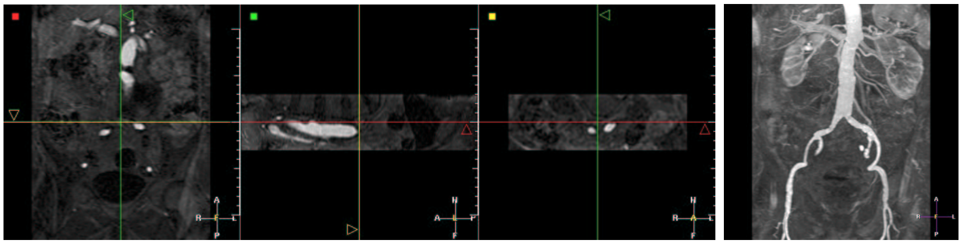


Figure 9.1. Contrast (Gd-DTPA) MRA image of the aorta: Left image shows three orthogonal views of the dataset; right image is a **MIP** view.

a vesselness function

$$\nu(s) = \begin{cases} 0, \text{ if } \lambda_2 \geq 0 \text{ or } \lambda_3 \geq 0 \\ (1 - \exp \frac{-{R_A}^2}{2\alpha^2}) \exp \frac{-{R_B}^2}{2\beta^2} (1 - \exp \frac{-S^2}{2c^2}) \text{ else} \end{cases}$$

where the ratios $R_A = \frac{|\lambda_2|}{|\lambda_3|}$ and $R_B = \frac{|\lambda_1|}{\sqrt{|\lambda_2 \lambda_3|}}$ are used to distinguish between lines and sheet-like structures and to measure deviation from blob-like structures. These measures arise from geometric interpretation as

$$\begin{aligned} R_A &= \frac{|\lambda_2|}{|\lambda_3|} = \frac{\pi|\lambda_2\lambda_3|}{\pi\lambda_3^2} = \frac{\text{largest cross-sectional area}/\pi}{(\text{largest axis length}/2)^2} \\ R_B &= \frac{|\lambda_1|}{\sqrt{|\lambda_2\lambda_3|}} = \frac{(4\pi/3)|\lambda_1\lambda_2\lambda_3|}{(4\pi/3)((1/\pi).\pi|\lambda_2\lambda_3|)^{3/2}} = \frac{\text{Volume}/(4\pi/3)}{\text{largest cross-sectional area}/\pi^{3/2}} \end{aligned}$$

and $S = \sqrt{{\lambda_1}^2 + {\lambda_2}^2 + {\lambda_3}^2}$ is used to reduce influence of the noise due to intensity variations in the background. See [60] for a detailed explanation of the settings of each parameter in this measure.

In figure 9.2 you can observe the response of the filter, based on the Hessian information, at three different scales: $\sigma = 1, 2, 5$. Using this information computed

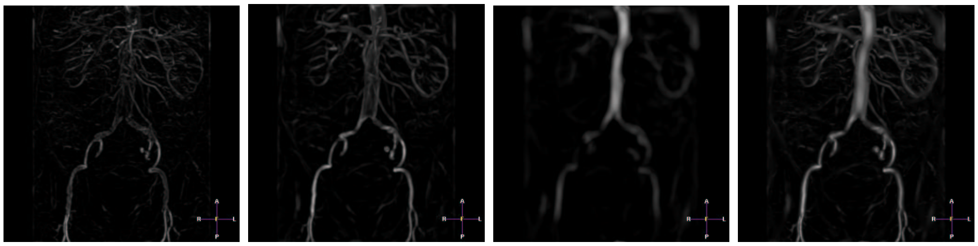


Figure 9.2. Ridge detection in the aorta image From left to right, the measure obtained at three different scales ($\sigma = 1, 2, 5$) and the maximum of the filter response across all scales (MIP visualization of the 3D images).

at several scales, the multiscale response of the filter is the maximum of the response of the filter across all scales, which is shown in figure 9.2-right.

Having computed this measure, we simply use it as speed function, thus propagating faster in the areas with higher filter response, taking

$$\|\nabla T(\mathbf{x})\| = \frac{1}{\max_{\sigma_{min} \le \sigma \le \sigma_{max}} \nu(\sigma, \mathbf{x}) + \nu_{min}} + w \tag{9.2}$$

where σ_{min} and σ_{max} are the minimum and maximum scales at which relevant structures are expected to be found, ν_{min} is just a constant which ensures that speed remains strictly positive, and w is the usual offset term introduced in [34]. This potential will be adapted to bright vascular structures on black background. This potential gives more information than the simple grey level value, since the filter response is higher in the center of the vessel.

9.1.2 Proposed solution

Initialization: *Freezing* method The use of the *Freezing* improves the resulting segmentation: Figure 9.3 shows the difference of segmentation obtained with (right image) and without pruning (middle image). Figure 9.3 demonstrates that the combination of multi-scale vessel enhancement and freezing enhances the segmentation ability of the *Fast-Marching*.

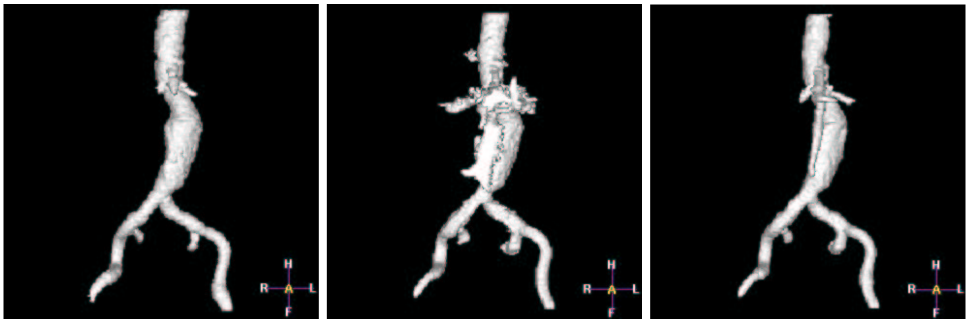


Figure 9.3. Comparing classical and freezing propagation in the Aorta: left image shows the resulting volume obtained using the *Fast-Marching* with a penalty model $\mathcal{P}(\mathbf{x}) = \max(I_{mean} - I(\mathbf{x}), 0)$ where I_{mean} is the mean value inside the aorta; middle image shows the result of a wave propagating in the Aorta MR dataset with a speed based on the Hessian eigenvalues; right image shows the same result using the *Freezing* approach of section 8.3.

Stopping: *Freezing* method This section is illustrated with figure 9.4[2]. Once the multiscale information is available, we can recall the same data than in the details on

[2] We would like to acknowledge Dr Wiro Niessen, from Image Sciences Institute, University Hospital Utrecht, Netherlands, who provided this image.

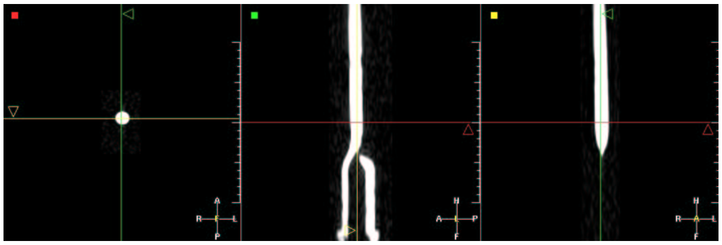


Figure 9.4. Phantom of a stenosed carotid artery in computed tomography angiography (CTA): This example has been chosen for illustration of the stopping criterion, by propagating from the top of the object with a speed $F = 1 \ (1 + \|\nabla I\|)$.

the stopping criterion in the previous chapter. If we plot the maximum distance d_{max} of section 8.3.1, across iterations while propagating, we will observe the following profile shown in figure 9.5. A great advantage of this multiscale information is that it can be used as a potential for obtaining centered trajectories, with no need to compute the distance to the object walls.

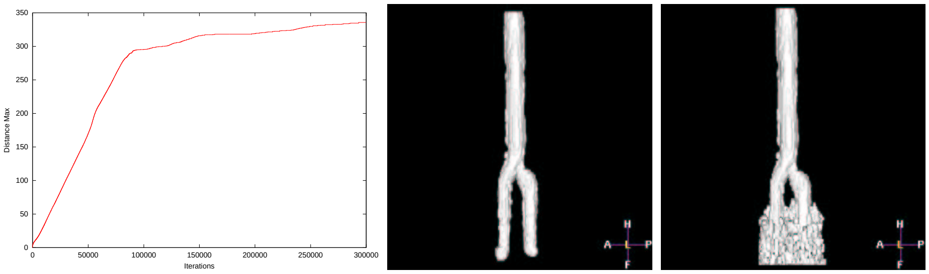


Figure 9.5. Detecting when the front floods outside the object: Left image is the plot of the maximum distance d_{max} across iterations; it is clearly visible that there is is an import change in the slope of the function around a distance maximum of 300; middle image represents the domain of voxels visited when $d_{max} = 290$, and right image is the same for $d_{max} = 310$.

9.1.3 Comparisons and conclusion of the tree extraction method

Other methods for skeletal representation

In [179], authors build a skeletal representation of an unorganized collection of scattered data points lying on a surface. They capture branching shapes, using a distance step similar to ours, by computing the k level-sets from the user-defined root of the tree; and for each of those level-sets, they extract the centroids of connected components. In our case it is not necessary to extract the centroids, because it introduces uncertainty in the location of the branching points. With a centering penalty $\mathcal{P}$, we *aggregate* the paths that are under a user-chosen distance d. This method based on the centroid extraction can be compared to the very interesting work of *Angella* found in [5, 6], which present a deformable and expansible tree as a skeleton extractor, where each node of the tree is a free particle that propagate into the data, pushed by repulsive forces coming from other particles and contours. The set of free particles describes the tree hierarchy. In our case, the sub-voxel precision is very important for visualization and measurements (see sections 7.1 and 7.2), and the needed number of particles for achieving this task would lead to huge computing times.

Very similar work can be found in [157], where the author use wavefronts to extract morphological descriptions of binary images, in particular binary tree structures. However, bifurcations are detected on a projected image in 2D, then this information is upgraded to 3D, but still the method is applied iteratively, looking for bifurcations at each iteration. Our method use a scale parameter c_d, as a distance step in our wavefront, only looking for bifurcations every time the front has crossed a multiple number of this distance. It reduces greatly computations, and can be parameterized by the user, who can only look for branches lower than a typical value d_{min} which is the upper-bound of our scale parameter.

Morphological techniques, like those in [157], are the main tool used for tree extraction, and lots of techniques, like thinning algorithms are already used in medical imaging. They start from volume images so that the traditional medial axis transform of *Blum* [15] can be applied, as in [130, 143]. However, the purpose of our application is to have a smooth set of multiple trajectories. This smoothness is needed for accurate measurements and visualization along the trajectories. Morphological techniques require post-processing in to remove undesirable small parts of the skeleton. Smoothing and removing undesirable small parts of the skeleton is done using our distance step and is very similar to techniques shown in [173], where the scale is also an input in the algorithm. To conclude with the use of morphological techniques, the skeletal description we are looking for corresponds to the needed of an accurate basis for observation and measurements of pathologies. We thus need a smooth and accurate information: a tree which describes the cylindrical topology of the object observed. The variation of the section of a tubular shapes leads to error in medial axis transforms, and to the need of post-processing techniques, to *clean* the skeleton obtained, that our method does not need.

Most impressive work on vascular quantification among others can be found in the PhD thesis of *Frangi*. He develop a very interesting method based on path and shape extraction in [58]:

- The author first set the two extremities of a path on the surface obtained through a iso-surface extraction process;

- the minimal path is extracted on the representation of the surface, using a technique similar to [90];

- a centering force, based on multi-scale enhancement filtering (see [60]) drives the minimal path in the center of the tube-shaped object;

- a circular cross section approximating the vessel is swept along the central vessel axis extracted previously (*swept surface*), and creates a deformable cylinder;
- this cylinder initiates a tensor product B-spline surface [142], that fits the boundaries of the vessel.

Using both path and shapes representation in the same framework, Frangi proposes an elegant method for quantification of vessel morphology [59].

In order to highlight important benefits of our method, we are going to compare our results, that are not completely dedicated to quantification of vascular diseases in MRA images. However, our method proposes an alternative that may overcome several drawbacks of his method.

Topology of the objects: In [57] the bifurcations in carotid arteries introduce errors in the measurements of the stenoses; with our method, bifurcations are localized and wrong measures near branching points can be omitted.

Branching points: We provide the measures in the whole set of branches of our objects, setting a unique tree root seed for segmentation and path extraction. In [57] *Frangi* gives the measure between the defined user end points (he gives also an interesting study of the variability of the results across the user initialization in [57]). In our case, only one point is needed. It enables to reconstruct the whole set of trajectories inside the object, but it converts this information into a tree hierarchy, where important information can be separated from the whole.

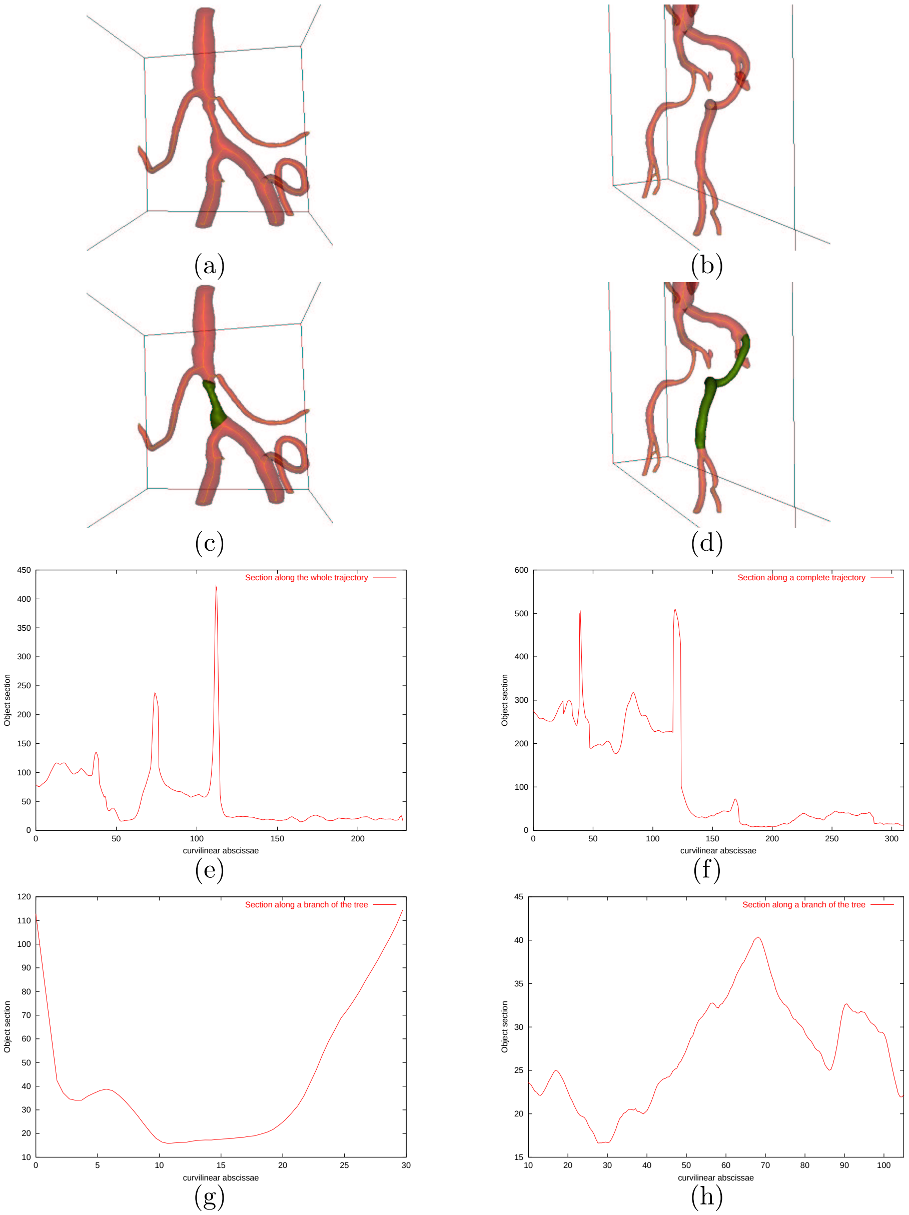
A result of this property of our method is shown in figure 9.6. In particular, figure 9.6-(g) is the information contained in the interval $[40;60]$ in figure 9.6-(e). It is the same process for figure 9.6-(h) which corresponds to the sub-plot contained in the interval $[190;250]$ in figure 9.6-(e). Therefore the tree extraction enables to localize accurately the information needed, as the stenosis extent for the case presented in the left column of figure 9.6.

Accuracy of the model the B-spline that extracts the vessel boundaries in [58] is an approximation of the surface, whereas the zero-level set embedded in $\tilde{\phi}$ has sub-pixel precision.

Conclusion on the vascular extraction method

We have finally a method which provides a sub-pixel information of the position of the shape. Based on the paths extracted with our fast and robust algorithm, the quantification rely on an accurate centered position of the path points. Thus measures and visualization are enhanced (see figure 9.7).

At a matter a fact, this visualization, once paths and shapes are extracted, is real-time, due to the fast rendering of the triangulation of our implicit model. Thus, camera trajectory is managed via the paths extracted. A further extension of this work could be to derive an interface to choose between each branch where to go inside the model.



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 9.6. Comparison of measurements on the tree and on a trajectory: Two different datasets are presented, each one in a column (left column dataset present a stenosed vessel). First row (a,b) displays segmented surfaces and extracted trees. Second row (c,d) displays the sub-volume of interest in both cases where sections are performed. Third row (e,f) shows plots of the section measured across the curvilinear abscissae of a trajectory. Fourth row (g,h) displays the same result using branches extracted between two bifurcations.

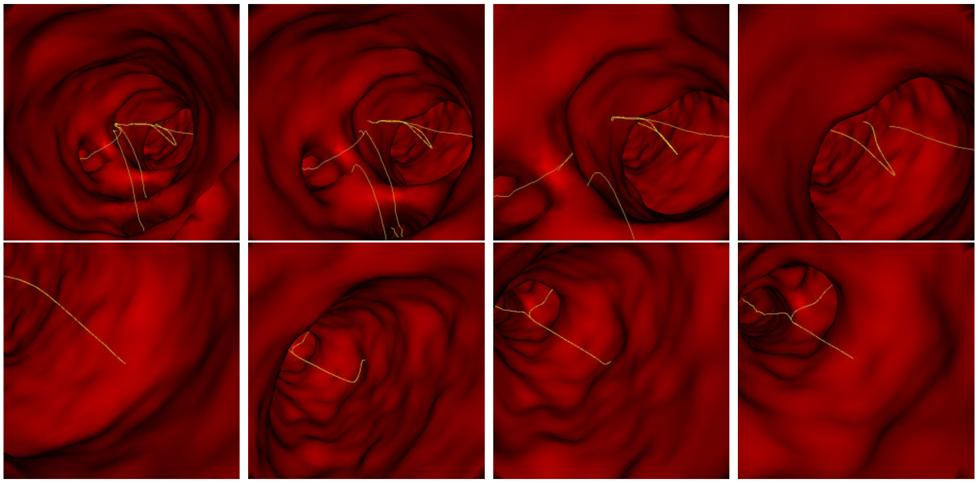


Figure 9.7. Endoscopic view along one trajectory: The whole set of trajectories is displayed (in yellow) simultaneously with the surface rendering: the user do not miss any bifurcation since he can see any branch.

9.2 Application to the Bronchial Tree

9.2.1 Medical interest

Extraction of the bronchi and the bronchial tree

First interest of this segmentation is to provide trajectories for a virtual bronchoscopy system, on the same basis than the virtual colonoscopy tool detailed in section 3.1.

A second important implementation possible is airway tree measurements that can be used to detect lesions or stenoses, structural abnormalities, and to evaluate airway reactivity to external stimuli (for example, evaluating asthma impact on the airways diameters). One must take into account the perspective distortion inherited from the volume rendering, for optimizing stent fitting for example, and the segmentation of the complex bronchi can provide an accurate information.

The framework we developed for the vascular tree extraction can be transposed to this different problem, in order to obtain both surfaces and centerlines of the airways, in typical datasets of the lungs, like multi-slice CT scanner, where voxels are nearly isotropic.

Role of the Virtual Bronchoscopy

The bronchoscopy technique has existed since 1897 and represents probably one of the most frequently used invasive procedures. Even in the hands of a clinically experienced pulmonologist, there is a risk for the patient. However, the goal of virtual bronchoscopy (see an example in [75]) is not to replace real bronchoscopy, which has high advantage of providing a direct inspection of the natural pigmentation which can clearly indicate a pathology. And inability to perform biopsies gives virtual bronchoscopy the role of a detection and much less a characterization technique.

Possible pathologies visible on the multi-slice CT scanner are for example tumors. Malignant tumors of the lung represent the most frequent cause of cancer death in males (35%) and female (18%). Since those tumors (benign or malignant) are only visible on the renderings system when they imply a morphological alteration of the bronchial wall, virtual bronchoscopy cannot contribute substantially to characterization of tumors. Thus, evaluation of virtual bronchoscopy should be restricted to the morphology of the bronchi and direct visualization of intraluminal masses.

However, one handicap of real endoscopy is its inability to see through the bronchial wall, whereas all information surrounding the object is available in virtual bronchoscopy. This handicap is very important since a clinician would like to plan a biopsy in a location that can be accessed only from the bronchial tree. In this case, the virtual bronchoscopy enables to determine in advance the optimal access point for the biopsy procedure.

A further indication for this process is the rare case when the real inspection is contraindicated, as in the presence of a strong stenosis of a branch, or as in the presence of an infiltration due to an extensive tumor manifestation, or in the case of an application in pediatrics, where the necessary sedation can be contraindicated.

Last important improvement brought by the virtual procedure is its clinician teaching device aspect [18, 125]: it can contribute th education and qualification of operating personnel (which benefits the patient by the way).

9.2.2 State of the art in Bronchoscopy imaging

Acquisition techniques

Computed Tomography represents the standard examination technique of the thoracic area, because a natural contrast exists between air and soft tissues, explaining why the trachea and the bronchial tree are perfectly suited for the generation of a virtual bronchoscopy. Three different types of CT data can be used:

1. Incremental CT: a slice is imaged in the axial orientation, after which the patient is shifted to the next position in order to image the next slice. 3D reconstruction (and hereafter renderings) can be calculated from incremental data only when the patient lies so still than no motion occurs during the whole examination. These data are of little use for virtual bronchoscopy;

2. Spiral CT: superior to incremental CT, the patient is shifted during the rotation of the tube-detector system. It enables to acquire large anatomical regions, like Thorax, in a single breath-hold. But still, the z-axis resolution is considerably worse than the in-plane resolution, and is a limiting factor for small bronchi.

3. Multi-slice CT: common systems image four slices at a time. This results in very low total acquisition time, but can also result in isotropic volume elements (voxels) in the final 3D dataset. Figure 9.8 displays a volume of interest of a classical multi-slice CT scanner of the lungs.

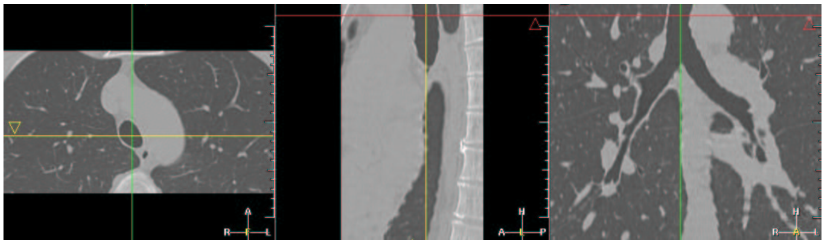


Figure 9.8. 3D Multi-Slice CT scanner of the Lungs: this is an isotropic volume of interest about 287 × 150 × 249 voxels.

Multi-slice CT is the ideal imaging modality for obtaining virtual bronchoscopic renderings of the smaller bronchi, since structures with diameter of 2mm should become clearly visible. However, the huge number of voxels can create a problem, as it overload the storage capacity of common workstations. Processing time in algorithms can also become so long than clinical application may not be realistic.

Segmentation techniques

They can be divided in three categories:

1. 2D methods: they can be completely manual, where the contours of the airway lumen is delineated manually in each axial images, but they strongly depend on the interobserver variability, and the loss of precision of the 2D segmentation towards the 3D information. They can also be semi-automatic, starting from an initial set of contours manually drawn, which is corrected and smoothed with detection algorithms, or by flooding gradient maps with region-growing in 2D [144]. However, those 2D methods work on the information contained in the cross-sections of the bronchi contains on not on the whole 3D data.

2. 3D methods: there is of course the usual set of 3D rendering techniques (**MPR** , **MIP** , volume or surface renderings). Other more elaborated methods are based on 3D region-growing algorithms. *Mori et al.* [124] detect an optimal threshold value in order to extract the airways, and once this segmentation is done, thin the airways to obtain the tree which is input in a recognition process for automatic labeling [123]. Those methods suffer from limitations due to the use of a threshold, and problems of accuracy for the small bronchi, since they provide a binary image as segmentation.

3. 2D to 3D methods: the principle is to segment the airways in each axial slice, and to reconstruct the 3D segmentation by combining the 2D segmentations, doing 3D/3D post-processing. Several techniques are based on the detection of the airways location in 2D [169]. But how important is the enhancement of the post-processing, the result depends on the 2D initialization, there is no pure 3D information involved in the detection. However, important improvements have been done in the field by *Fetita* [52], as well in 2D/3D as in pure 3D.

We have worked upon the use of our method involving both *Fast-Marching* and *Level-Sets* methods to extract the airways, and the airway tree. We encountered several problems due to the specificity of the bronchi in CT, during the initialization step. All tests are detailed in the following.

9.2.3 Applying our framework

Initialization: Region growing

Several problems exist with the use of the *Fast-Marching* algorithm as a region growing method for airway segmentation. Since it relies on the edge strength of the airway walls that weak at several places, the propagating front *leaks* out into the surrounding parenchyma. We have already seen that the *Fast-Marching* can flood into the surrounding pixels of the tubular structures, and we have built a method based on *Freezing* pixels (see section 8.3.1) in order to avoid *leakage*. This method was successfully applied for vascular tree extraction in chapter 8, using as speed function a multiscale filtering technique derived from work in [60]. But in the case of the lungs airway, the grey level information is very different, as shown in figure 9.9. In figure 9.9-right, we observe that the minima occur near the center of the airways and the maxima occur near the middle of the walls. But this information is relatively poor, and partial volume effects can occur: as the diameter of the airways decreases, partial volume averaging begins to increase the value within the lumen, and the *Fast-Marching* will flood the parenchyma at a weak wall, as shown in figure 9.10. The *Fast-Marching* algorithm is applied in this case with a potential based on the grey level information. Using $\mathcal{P}(\mathbf{x}) = \max\left(I(\mathbf{x}) - I_{airways}, 0\right) + w$, with $I_{airways}$ being an approximate value corresponding to air, cannot provide a result where the propagation critically depends on the weakness of the edges. Using the grey level information for the lungs is similar to using the gradient in the vascular contrast enhanced medical images: it is not valuable. The profile of the airway in figure 9.9-right is somehow similar to a “Mexican hat”, but the Hessian information given by a measure based on its eigenvalues will detect the inner and outer walls of the airways, and will not give

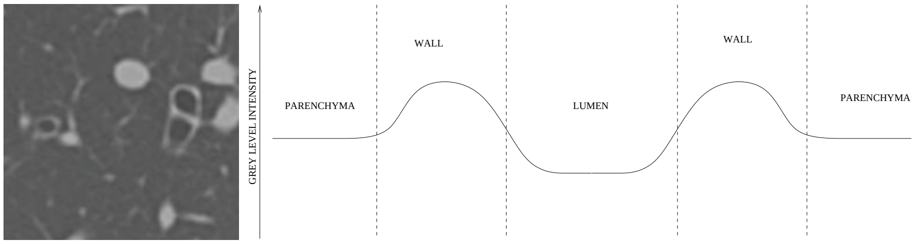


Figure 9.9. Profile of the airway grey level: Left image is a zoom on a 2D axial slice of the multi-slice reformat image of the lungs, we can distinguish the lumen in the black circle areas surrounded by the airway walls in bright intensities; right image is an illustration of the profile of the grey level intensity, along a line crossing through an airway center.

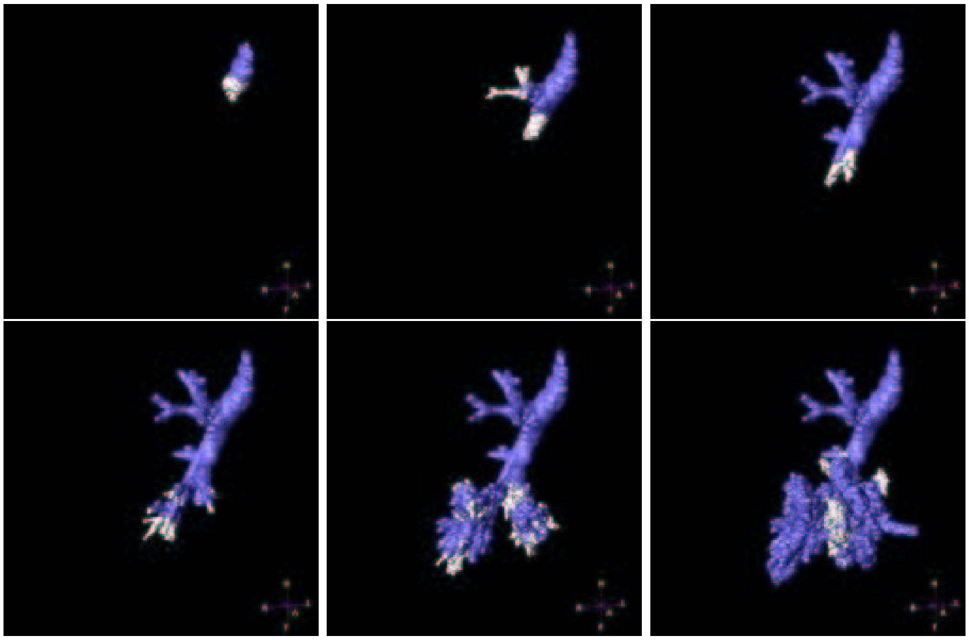


Figure 9.10. Flooding of the *Fast-Marching* in the parenchyma: the images are samples of the front propagation process in the lungs airways, with the *Freezing* methodology; the frozen pixels are represented in blue whereas the propagating parts are in white.

a high response in the lumen. Since then, this problem defines the limits of the use of the *Fast-Marching* in pre-segmentation, as well as with the use of a more complicated heuristic such as the *Freezing* algorithm.

However, there are other ways to create a pre-segmentation which can be input in a *Level-Sets* model. Among others, we decided to use methods developed by *Mori et al.* in [124]. The principle of this initialization is based on the reason of the failure of the *Fast-Marching* method: it focuses on the detection of the *flooding* inside the parenchyma. It extracts the inside of the area of the bronchus by tracing voxels with relatively small CT values corresponding to air without processing across voxels with relatively large CT values, assuming that the airway area is simply connected.

The method is a simple region-growing in the 3D image, starting from a point inside the trachea. This point, which will be the root of the final tree hierarchy can be easily detected in the 3D dataset, as shown in figure 9.8

The algorithm is the following:

Definition

- a start point $\mathbf{x}_0$: the region growing needs a seed point for starting. In order to set a protocol of segmentation, the start point needs to be always initialized in the same region, inside the trachea, before the first bifurcation in the tracheobronchial tree. Luckily enough, the trachea can be recognized stably in the multi-slice CT dataset.

Moreover this seed point will be the root of the tree hierarchy extracted at the end of the whole process, which enables to reduce user interaction to the setting of the seed point only.

- an initial threshold value I_0, a threshold value $I_{threshold}$ with a threshold step dI, for the original volume image I
- a segmentation defined by a binary mask $\mathcal{M}$ where $\mathcal{M}(\mathbf{x}) = 0$ if $\mathbf{x}$ is inside the object, and $\mathcal{M}(\mathbf{x}) = -1$ elsewhere.

Initialization

- $I_{threshold} = I_0$;
- $\mathcal{M}(\mathbf{x}_0) = 0$ and $\mathcal{M}(\mathbf{x}) = -1$ elsewhere;

Loop:

- at each iteration i, we binarize image I, defining the mask I_B where $I_B(\mathbf{x}) = 0$ if $I(\mathbf{x}) < I_{threshold}$, $I_B(\mathbf{x}) = -1$ elsewhere;
- we apply a connectivity algorithm, to connect to the pixels $\mathbf{x}$ that verify $\mathcal{M}(\mathbf{x}) = 0$ all voxels $\mathbf{y}$ with $I_B(\mathbf{y}) = 0$.
- For all voxels $\mathbf{y}$ connected, $\mathcal{M}(\mathbf{y}) = 0$;
- We count N_i the total number of voxels $\mathbf{x}$ with $\mathcal{M}(\mathbf{x}) = 0$ at iteration i (see figure 9.11);
- If $N_i > N_{max}$, the optimal threshold is $I_{threshold}$ and we stop;
- $I_{threshold} = I_{threshold} + dI$.

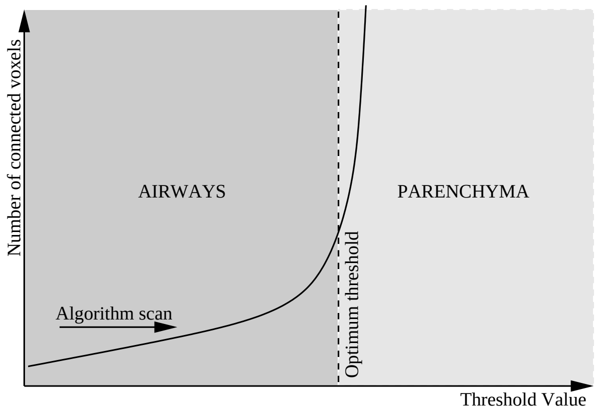


Figure 9.11. Detection of the optimal threshold value: When the threshold value is increased, the number of connected voxels below this threshold increases, until it reaches the optimal value; a superior threshold will lead to the flooding in the parenchyma.

When the number of segmented voxels exceeds the area threshold value N_{max}, the algorithm is stopped, just before explosion in the number of voxels visited occurs in figure 9.11.

Final Segmentation and Automatic Branch extraction

Since the initialization is obtained, the inside and outside regions are used in order to initialize the region-based forces of the *Level-Sets* model. In the case of the lungs, we used the sigmoidal region-based forces formulation for several reasons:

- the computing time is very important, considering the size of the dataset, and the use of constant forces across time reduces this cost;
- the parenchyma has a distribution very similar to that of the airways. The use of region-based forces should then induce the use of three different regions: one for the lumen, one for the parenchyma, and one for the soft tissues (and others). Managing three regions will greatly increase the computing time.
- if we use two regions instead of three, the parenchyma is contained in the outside region and reduces greatly the mean and increases the variance of the model. There is a huge risk that the outside has a variance too important, and shrinks the lumen segmented. This problem has been already presented in the application concerning the visualization of the colon polyps.

Several iterations are necessary in order to extract the lower bronchi, as shown in figure 9.12. Once this segmentation step is achieved, we extract the trajectories from the starting point $\mathbf{x}_0$, and we convert it into a tree, using the labeling methods of section 8.4. We optimize this extraction, in function of the length of the minimal branch to be extracted, as shown in figure 9.13, where we display the label map for several minimal length. Since this step represents a computing cost, this minimal length must be accurately set according to the needs of the clinician (since this length shrinks with the depth in the bronchial tree).

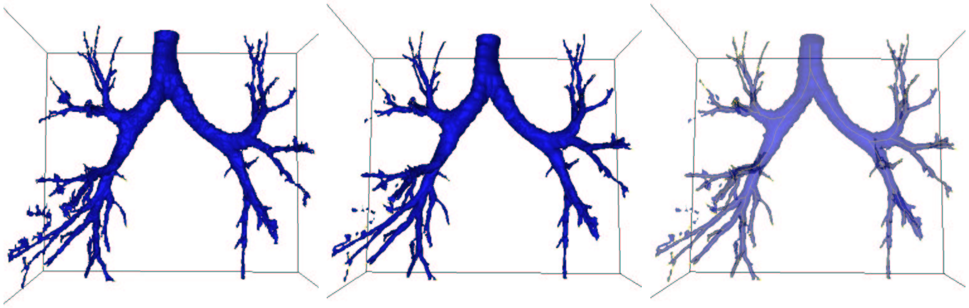


Figure 9.12. Three steps of the airways tree segmentation: Left image is the initialization given by the method [124] described in section 9.2.3; middle image is the surface of the airways after 40 iterations of the *Level-Sets* model; right image shows the whole tree extracted in the airways with the labeling algorithm illustrated in figure 9.13.

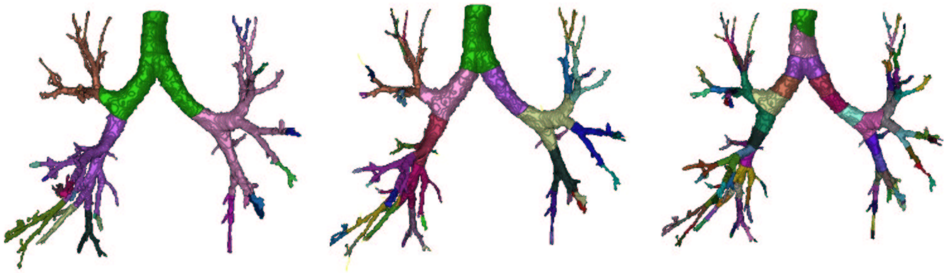


Figure 9.13. Labeling of the airways: Using the method described in section 8.4, we label the airways segmented, according to a chosen distance step; from left to right are shown the label maps for respective steps 100, 50 and 20.

9.2.4 Conclusion

Artifacts

Even with multi-slice technology, one problem remains unsolved: cardiac motion cannot be suppressed, despite fast acquisition. Only a trigger, as employed for cardiac imaging could provide assistance. This motion extends to the neighboring pulmonary parenchyma, and results in irregularities in the bronchial wall (see figure 9.14-left). The impact of this motion onto the segmentation is not clear, but it seems that the surface extracted near the hear in the lower left part of the bronchi is not correctly segmented (see figure 9.14-middle). These artifacts can be reduced by doing multi-slice CT acquisition with a trigger, as done for cardiac imaging, taking into account

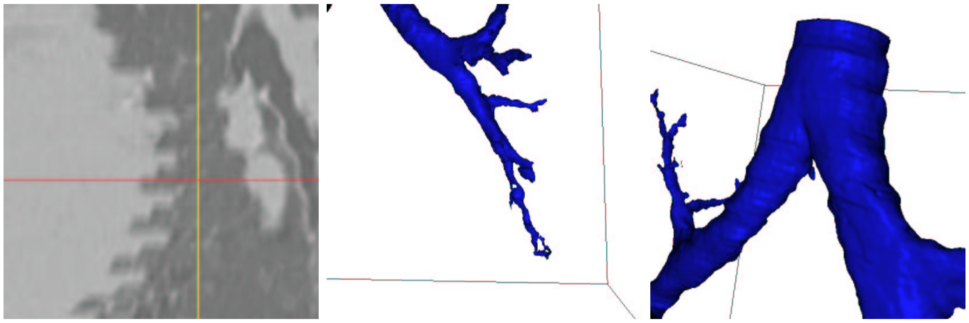


Figure 9.14. Two kind of artifacts in the multi-slice datasets: Left image shows the motion of the heart in a sagittal slice of the 3D dataset of figure 9.8 during acquisition; middle image shows the resulting poor segmentation of the lower-left part of the airways; right image displays the pulsation artifacts carried over from the aorta onto the trachea: these rings should not be confused with the natural tracheal cartilage.

cardiac pulsation for the time of acquisition.

Much more striking are artifacts produced by the aortic pulsations, transmitted to the trachea and the main bronchi: they appear as ring-like structures on the 3D surface, similarly to cartilage rings. However, they can easily be distinguished from the tracheal cartilages, since they appear horizontal in the lice direction (see figure 9.14-right). Those rings imply errors in the measures of the airways diameters, but there is no particular possibility to avoid them, since a correlation with the heart pulsations exists but is difficult to model.

Perspectives

Results are impressive, and we can easily obtain a virtual bronchoscopic view, using the tree structure to guide the virtual endoscope and the triangulated surface, obtained through th *Marching-Cubes* algorithm with the zero-level set of our level-set function (see figure 9.15)

However, there are several technical improvements that are currently missing

- Improving fast-marching for the airway initialization: the method developed has failed in giving an accurate initialization. The complex structure of the object, and the thin-walled bronchi plus the partial volume effect lead to wrong results. One possible extension could be to modify the speed function, adapting it to the depth of the current voxels involved in the computation, since the partial volume effect increases with the depth in the airway tree.

- Reducing the number of iterations of the *Level-Sets* model: 40 iterations is still a huge number if each iterations is computed on the whole volume.

- Improving the *Level-Sets* model: the smaller parts of the bronchi are not recovered, the region-based formulation is not dedicated to the extraction of the thin curves. Using co-dimension 2 geodesic contours, as done by *Lorigo et al.* [108] for vessels in MRA images, and using other expression of the flows like *Vasilevskiy and Siddiqi* [177].

- Developments: *Mori* was using its tree extraction method for automatic labeling of the bronchial tree [123]. If it is possible to assign the anatomical names to the bronchial branches extracted from CT images and to display the name of the currently observed branch on a virtual bronchoscopy image, it will help clinicians to understand the current observing position. This tree structure can also be input in a system to assist biopsies in the tracheobronchial tree, as done in [17].

- Validation: still, clinical evaluation of the method is not done, and in particular the evaluation of the tree structure extracted should be evaluated with the choice of the minimal length of the branches extracted.

- Benchmark: a possibility of a benchmark with the tremendous work of *Fetita* [52, 144] has been scheduled but is not achieved yet.

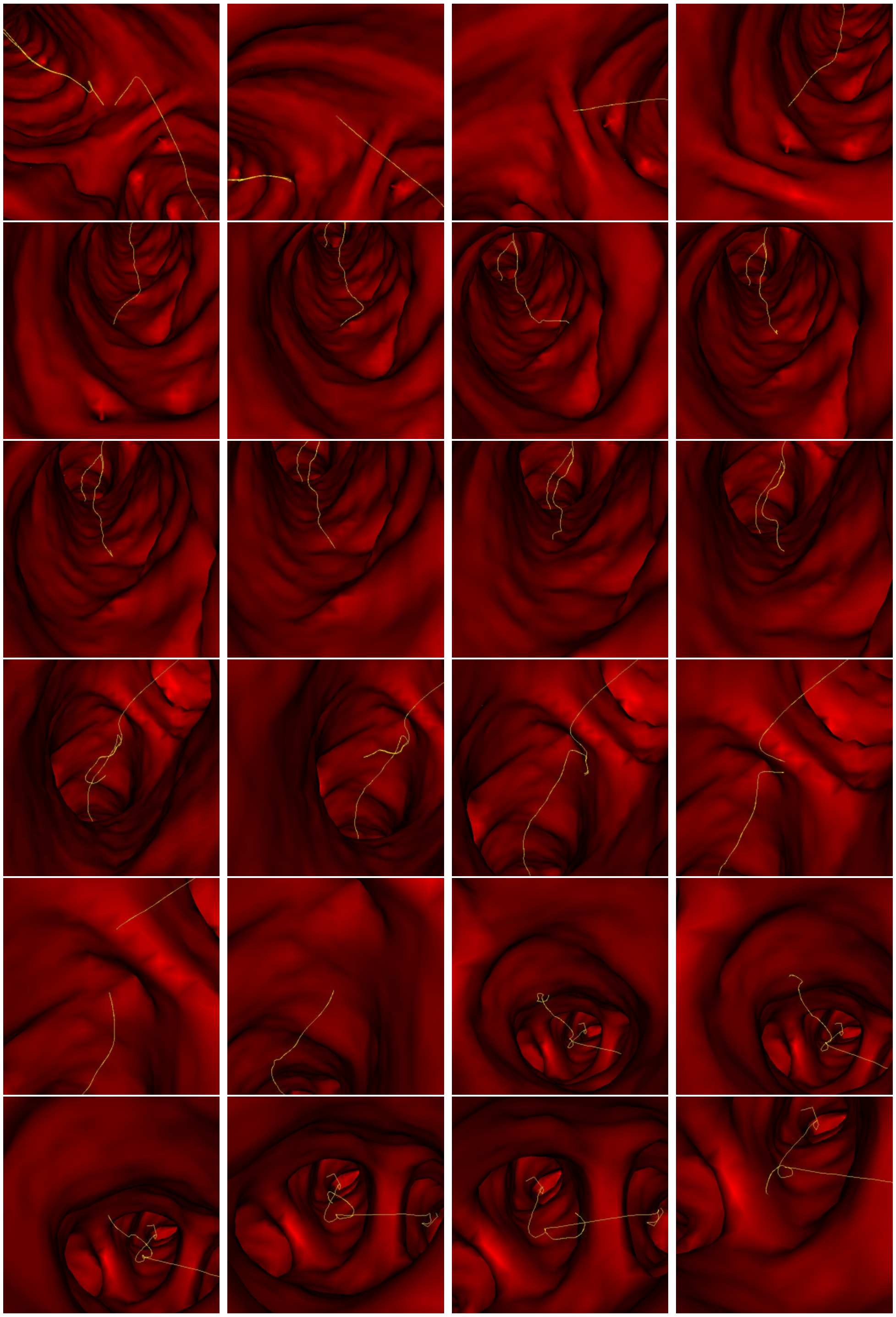


Figure 9.15. Virtual Bronchoscopy: the images are samples of a movie automatically generated with the surface and the tree extracted in the 3D dataset with our Path and Shape extraction framework.

9.3 Reconstruction of vessels in 2D and 3D images using *Perceptual Grouping*

Since their introduction, active contours [82] have been extensively used to find the contour of an object in an image through the minimization of an energy. In order to get a set of contours with T-junctions, we need many active contours to be initialized on the image. The level sets paradigm [113, 23] allows changes in topology. It enables to get multiple contours by starting with a single one. However, it does not give satisfying results when there are gaps in the data since the contour may propagate into a hole and then split into several curves where only one contour is desired. This is the problem encountered with *Perceptual Grouping* when we try to group a set of incomplete contours. For example, in a binary image like in figure 9.16 with a drawing of a shape with holes, human vision can easily fill in the missing boundaries and form complete curves. *Perceptual Grouping* is an old problem in computer vision. It has been approached more recently with energy methods [164, 72, 187]. These methods find a criteria for saliency of a curve component or for each point of the image. This saliency measure is based indirectly on a second order regularization snake-like energy ([82]) of a path containing the point. However, the final curves are generally obtained in a second step as ridge lines of the saliency criteria after thresholding. Motivated by this relationship between energy minimizing curves like snakes and completion contours, we worked upon finding a set of completion contours on an image as a set of energy minimizing curves.

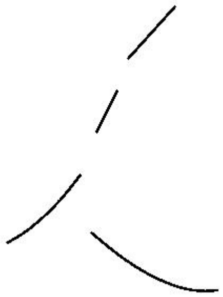


Figure 9.16. Examples of connected regions to be completed: The four regions are the four black components, on a bright background.

In order to solve global minimization for snakes, *Cohen and Kimmel* [34] used the minimal paths, as introduced in [87, 86]. The goal was to avoid local minima without demanding too much on user initialization, which is a main drawback of classic snakes [29]. Only two end points were needed. The numerical method has the advantage of being consistent (see [34]) and efficient using the Fast Marching algorithm introduced in [161]. In [31], the author proposed a way to use this minimal path approach to find a set of curves drawn from a set of points in the image. We also introduced a technique that automatically finds a set of *key* end points. In this chapter, we extend the previous approach to connected components instead of end points. In order to obtain a set of most salient contour curves, we find a set of minimal paths between pairs of connected components.

This approach is then extended for application in the completion of tube-like structures in 2D and 3D images. The problem is here to complete a partially detected object, based on some detected connected components that belong to this object.

For *Perceptual Grouping* , the potential P to be minimized along the curves is usually an image of edge points that represent simple incomplete shapes, as in figure 9.16. These edge points are represented as a binary image with small potential values along the edges and high values at the background. The potential could also be defined as edges weighted by the value of the gradient or as a function of an estimate of the gradient of the image itself, $P = g(\|\nabla I\|)$, like in classic snakes. The potential could also be a grey level image as in [34]. It could also be a more complicated function of the grey level. In our real examples of vascular structures in 2D and 3D, we use a potential based on a vesselness filter [60].

We present in Section 9.3.1 how to find a set of curves from a given set of unstructured points. Grouping the points in connected components, we propose a way to find the pairs of linked connected components and the paths between them. We then extend this approach to 3D and show an application in 3D medical images.

9.3.1 Finding Contours from a Set of Connected Components

Minimal Path between two Regions

The method of [34], detailed in the previous section allows to find a minimal path between two endpoints. This is a straightforward extension to define a minimal path between two regions of the image. Given two connected regions of the image $\mathbf{R}_0$ and $\mathbf{R}_1$, we consider $\mathbf{R}_0$ as the starting region and $\mathbf{R}_1$ as a set of end points. The problem is then finding a path minimizing energy among all paths with start point in $\mathbf{R}_0$ and end point in $\mathbf{R}_1$. The minimal action is now defined by

$$U(p) = \inf_{\mathcal{A}_{\mathbf{R}_0,p}} E(C) = \inf_{p_0 \in \mathbf{R}_0} \inf_{\mathcal{A}_{p_0,p}} E(C) \tag{9.3}$$

where $\mathcal{A}_{\mathbf{R}_0,p}$ is the set of all paths starting at a point of $\mathbf{R}_0$ and ending at p. This minimal action can be computed the same way as before in table 2.1, with the alive set initialized as the whole set of points of $\mathbf{R}_0$, with $U = 0$ and trial points being the set of 4-connexity neighbors of points of $\mathbf{R}_0$ that are not in $\mathbf{R}_0$. Back-propagation by gradient descent on U from any point p in the image will give the minimal path that join this point with region $\mathbf{R}_0$.

In order to find a minimal path between region $\mathbf{R}_1$ and region $\mathbf{R}_0$, we determine a point $p_1 \in \mathbf{R}_1$ such that $U(p_1) = \min_{p \in \mathbf{R}_1} U(p)$. We then back-propagate from p_1 to $\mathbf{R}_0$ to find the minimal path between p_1 and $\mathbf{R}_0$, which is also a minimal path between $\mathbf{R}_1$ and $\mathbf{R}_0$.

Minimal Paths from a Set of Connected Components

We are now interested in finding many or all contours in an image. We assume that from some preprocessing, or as data, we have an initial set of contours. We denote $\mathbf{R}_k$ the connected components of these contours. We propose to find the contours as a set of minimal paths that link pairs of regions among the $\mathbf{R}_k$'s. If we also know which pairs of regions have to be linked together, finding the whole set of contours is a trivial application of the previous section. The problem we are interested in here is also to find out which pairs of regions have to be connected by a contour. Since the set of contours $\mathbf{R}_k$'s is assumed to be given unstructured, we do not know in advance how the regions connect. This is the key problem that is solved here using a minimal action map.

Method

Our approach is similar to computing the distance map to a set of regions and their Voronoi diagram. However, we use here a weighted distance defined through the potential P. This distance is obtained as the minimal action with respect to P with zero value at all points of regions $\mathbf{R}_k$. Instead of computing a minimal action map for each pair of regions, as in Section 9.3.1, we only need to compute one minimal action map in order to find all paths. At the same time the action map is computed we determine the pairs of regions that have to be linked together. This is based on finding meeting points of the propagation fronts. These are *saddle points* of the minimal action U. These saddle points were already used for closed boundary extraction in [34] In Section 1.1.2, we said that calculation of the minimal action can be seen as the propagation of a front through equation (1.5). Although the minimal action is computed using fast marching, the level sets of U give the evolution of the front. During the fast marching algorithm, the boundary of the set of alive points also gives the position of the front. In the previous section, we had only one front evolving from the starting region $\mathbf{R}_0$. Since all points p of regions $\mathbf{R}_k$ are set with $\mathcal{U}(p) = 0$, we now have one front evolving from each of the starting regions $\mathbf{R}_k$. In what follows when we talk about front meeting, we mean either the geometric point where the two fronts coming from different $\mathbf{R}_k$'s meet, or in the discrete algorithm the first alive point which connects two components from different $\mathbf{R}_k$'s (see Figures 9.17 and 9.18).

We use the fact that given two regions $\mathbf{R}_1$ and $\mathbf{R}_2$, the saddle point $\mathbf{s}$ where the two fronts starting from each region meet can be used to find the minimal path between $\mathbf{R}_1$ and $\mathbf{R}_2$. Indeed, the minimal path between the two regions has to pass by the meeting point $\mathbf{s}$. This point is the point half way (in energy) on a minimal path between $\mathbf{R}_1$ and $\mathbf{R}_2$. Back-propagating from $\mathbf{s}$ to $\mathbf{R}_1$ and then from $\mathbf{s}$ to $\mathbf{R}_2$ gives the two halves of the path.

Notations and definitions

Here are some definitions that will be used in what follows. X being a set of points in the image, U_X is the minimal action obtained by Fast Marching with potential $\tilde{P}$ and starting points $\{p, p \in X\}$. This means that all points of X are initialized as

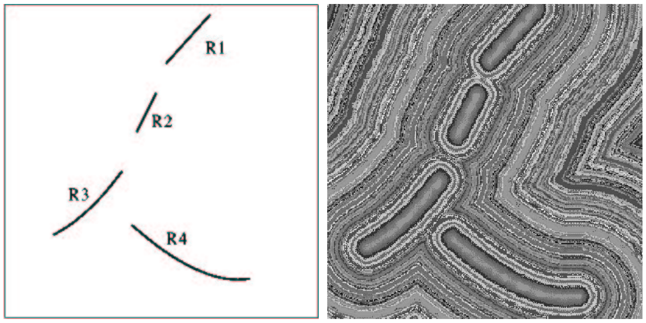


Figure 9.17. Minimal Action map from the four regions of the example of figure 9.16: On the right with a random LUT to show the level sets.

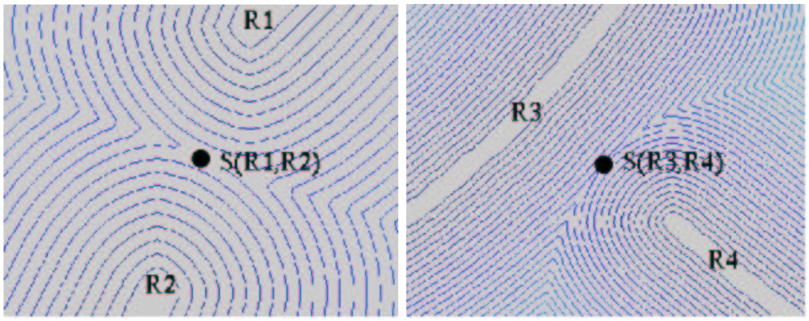


Figure 9.18. Zoom on *saddle points* **between regions:** Left image shows the iso-action levels near the saddle point between $\mathbf{R}_1$ and $\mathbf{R}_2$

alive points with value 0. All their 4-connexity neighbors that are not in X are *trial* points. This is easy to see that $U_X = \min_{p \in X} U_p$. X may be a connected component R or a set of connected components.

The *label* l at a point p is equal to the index k of the region $\mathbf{R}_k$ for p closer in energy to $\mathbf{R}_k$ than to other regions $\mathbf{R}_j$. This means that minimal action $U_{\mathbf{R}_k}(p) \leq U_{\mathbf{R}_j}(p), \forall j \neq k$. We define the region $L_k = \{p / l(p) = k\}$. If $X = \cup_j \mathbf{R}_j$, we have $U_X = U_{\mathbf{R}_k}$ on L_k and the computation of U_X is the same as the simultaneous computation of each $U_{\mathbf{R}_k}$ on each region L_k. These are the simultaneous fronts starting from each $\mathbf{R}_k$.

A *saddle point* $\mathbf{s}(\mathbf{R}_i, \mathbf{R}_j)$ between $\mathbf{R}_i$ and $\mathbf{R}_j$ is the first point where the front starting from $\mathbf{R}_i$ to compute $\mathcal{U}_{\mathbf{R}_i}$ meets the front starting from $\mathbf{R}_j$ to compute $\mathcal{U}_{\mathbf{R}_j}$; At this point, $U_{\mathbf{R}_i}$ and $U_{\mathbf{R}_j}$ are equal and this is the smallest value for which they are equal.

Two different regions among the $\mathbf{R}_k$'s will be called *linked regions* if they are selected to be linked together. The way we choose to link two regions is to select some *saddle points*. Thus regions $\mathbf{R}_i$ and $\mathbf{R}_j$ are *linked regions* if their *saddle point* is among the selected ones.

A *cycle* is a sequence of different regions $\mathbf{R}_k, 1 \leq k \leq K$, such that for $1 \leq k \leq K-1$, $\mathbf{R}_k$ and $\mathbf{R}_{k+1}$ are *linked regions* and $\mathbf{R}_K$ and $\mathbf{R}_1$ are also *linked regions*.

Finding and Selecting Saddle Points

The main goal of our method is to obtain all significant paths joining the given regions. However, each region should not be connected to all other regions, but only to those that are closer to them in the energy sense. There are many possibilities for deciding which regions connect together depending on the kind of data and application. In some cases, the goal would be to detect closed curves and avoid forming branches, as in [31]. Then the criterion would be to constrain a region to be linked to at most two other regions in order to make *cycles*. In our context, we are interested in detecting branches and avoiding closed curves. Therefore the criterion for two regions $\mathbf{R}_i$ and $\mathbf{R}_j$ to be connected is that their fronts meet without creating a "cycle".

We see in Figure 9.18 a zoom on the saddle points detected between regions $\mathbf{R}_1$ and $\mathbf{R}_2$ and $\mathbf{R}_3$ and $\mathbf{R}_4$. Once a *saddle point* $\mathbf{s}(\mathbf{R}_i, \mathbf{R}_j)$ is found and selected, back-propagation relatively to final energy U should be done both ways to $\mathbf{R}_i$ and to $\mathbf{R}_j$ to find the two halves of the path between them. We see in Figure 9.19 this back-propagation at each of the three automatically selected *saddle points*. They link $\mathbf{R}_1$ to $\mathbf{R}_2$, $\mathbf{R}_2$ to $\mathbf{R}_3$ and $\mathbf{R}_3$ to $\mathbf{R}_4$. At a saddle point, the gradient is zero, but the direction of descent towards each point are opposite. For each back-propagation, the direction of descent is the one relative to each region. This means that in order to estimate the gradient direction toward $\mathbf{R}_i$, all points in a region different from L_i have their energy put artificially to ∞. This allows finding the good direction for the gradient descent towards $\mathbf{R}_i$. However, as mentioned earlier, these back-propagations have to be done only for selected *saddle points*. In the fast marching algorithm we have a simple way to find *saddle points* and update the *linked regions*.

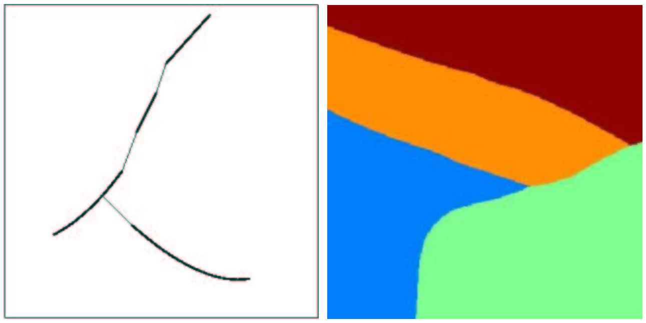


Figure 9.19. Example with four regions: On the left we show the minimal paths obtained by back-propagation from the three *saddle points* to each of the regions from where the front comes; on the right, and the Voronoi diagram obtained.

As defined above, the region L_k associated with a region $\mathbf{R}_k$ is the set of points p of the image such that minimal energy $U_{\mathbf{R}_k}(p)$ to $\mathbf{R}_k$ is smaller than all the $\mathcal{U}_{\mathbf{R}_j}(p)$ to other regions $\mathbf{R}_j$. The set of such regions L_k covers the whole image, and forms the Voronoi diagram of the image (see figure 9.19). All *saddle points* are at a boundary between two regions L_k. For a point p on the boundary between L_j and L_k, we have $\mathcal{U}_{\mathbf{R}_k}(p) = U_{\mathbf{R}_j}(p)$. The *saddle point* $\mathbf{s}(\mathbf{R}_k, \mathbf{R}_j)$ is a point on this boundary with minimal value of $U_{\mathbf{R}_k}(p) = U_{\mathbf{R}_j}(p)$. This gives us a rule to find the *saddle points* during the fast marching algorithm.

Each time two fronts coming from $\mathbf{R}_k$ and $\mathbf{R}_j$ meet for the first time, we define the meeting point as $\mathbf{s}(\mathbf{R}_k, \mathbf{R}_j)$. This means that we need to know for each point of the image from where it comes. This is easy to keep track of its origin by generating an index map updated at each time a point is set as alive in the algorithm. Each point of region $\mathbf{R}_k$ starts with label k. Each time a point is set as alive, it gets the same label as the points it was computed from in formula (1.7). In that formula, the computation of $U_{i,j}$ depends only on at most two of the four pixels involved. These two pixels, said A_1 and B_1, have to be with the same *label*, except if (i, j) is on the boundary between two labels. If A_1 and B_1 are both alive and with different labels k and l, this means that regions $\mathbf{R}_k$ and $\mathbf{R}_l$ meet there. If this happens for the first time, the current point is set as the *saddle point* $\mathbf{s}(\mathbf{R}_k, \mathbf{R}_l)$ between these regions. A point on the boundary between $\mathbf{R}_k$ and $\mathbf{R}_l$ is given the label of the neighbor point with smaller action A_1. At the boundary between two labels there can be a slight error on labeling. This error of at most one pixel is not important in our context and could be refined if necessary.

Algorithm

The algorithm for this section is described in Table 9.1 and illustrated in figures 9.17 to 9.19. When there is a large number of $\mathbf{R}_k$'s, this does not change much the computation time of the minimal action map, but this makes more complex dealing with the list of linked regions and *saddle points* and testing for *cycles*.

The way we chose to test for cycles is as follows. Assume a saddle point between regions $\mathbf{R}_i$ and $\mathbf{R}_j$ is found. We then test if there is already a link between these regions through other regions. This means we are looking for a sequence of different regions $\mathbf{R}_k, 1 \leq k \leq K$, with $\mathbf{R}_1 = \mathbf{R}_i$ and $\mathbf{R}_K = \mathbf{R}_j$, such that for $1 \leq k \leq K-1$, $\mathbf{R}_k$ and $\mathbf{R}_{k+1}$ are *linked regions*.

This kind of condition can be easily implemented using a recursive algorithm. When two regions $\mathbf{R}_i$ and $\mathbf{R}_j$ are willing to be connected - i.e. that their fronts meet - a table storing the connectivity between each region enables to detect if a link already exists between those regions. Having N different regions, we fill a matrix $M(N, N)$ with zeros, and each time two regions $\mathbf{R}_i$ and $\mathbf{R}_j$ meet without creating a cycle, we set $M(i, j) = M(j, i) = 1$. Thus, when two regions meet, we apply the algorithm detailed in table 9.2.

If two regions are already linked, the pixel where their fronts meet is not considered as a valuable candidate for back-propagation. The algorithm stops automatically when all regions are connected.

Minimal paths between Regions $\mathbf{R}_k$

- Initialization:
  - $\mathbf{R}_k$'s are given
  - $\forall k, \forall p \in \mathbf{R}_k, V(p) = 0; l(p) = k;$ p *alive*.
  - $\forall p \notin \cup_k \mathbf{R}_k, V(p) = \infty; l(p) = -1;$ p is *far* except 4-connexity neighbors of $\mathbf{R}_k$'s that are *trial* with estimate U using equation (1.7).
- Loop for computing $V = U_{\cup_k \mathbf{R}_k}$:
  - Let $p = (i_{min}, j_{min})$ be the *Trial* point with the smallest action U;
  - Move it from the *Trial* to the *Alive* set with V(p) = U(p);
  - Update $l(p)$ with the same index as point A_1 in formula (1.7). If $R(A_1) \neq R(B_1)$ and we are in case 1 in table 2.2 where both points are used and if this is the first time regions of labels $l(A_1)$ and $l(B_1)$ meet, $\mathbf{s}(\mathbf{R}_{l(A_1)}, \mathbf{R}_{l(B_1)}) = p$ is set as a *saddle point* between $\mathbf{R}_{l(A_1)}$ and $\mathbf{R}_{l(B_1)}$. If adding a link between these regions does not create a *cycle*, they are set as *linked regions* and $\mathbf{s}(\mathbf{R}_{l(A_1)}, \mathbf{R}_{l(B_1)}) = p$ is selected,
    For each neighbor (i, j) of (i_{min}, j_{min}):
    - ∗ If (i, j) is *Far*, add it to the *Trial* set;
    - ∗ If (i, j) is *Trial*, update action $U_{i,j}$.
- Obtain all paths between selected *linked regions* by back-propagation each way from their *saddle point* (see Section 9.3.1).

Table 9.1. Algorithm of Section 9.3.1

Application

The method can be applied to connected components from a whole set of edge points or points obtained through a preprocessing. Finding all paths from a given set of points is interesting in the case of a binary potential defined, like in Figure 9.17, for *Perceptual Grouping* . It can be used as well when a special preprocessing is possible, either on the image itself to extract characteristic points or on the geometry of the initial set of points to choose more relevant points. We show in figures 9.20 and 9.21 an example of application to a medical image of the hip where the objects of interest are the vessels. Potential P is defined using ideas from [60] on vesselness filter (detailed later in section 9.3.2). About vessel detection, see also [177, 107].

9.3.2 Finding a Set of Paths in a 3D Image

Extension to 3D

We now extend our approach to finding a set of 3D minimal paths between regions in 3D images. All definitions and algorithms of section 9.3.1 are not affected by changing the dimension of the image from 2D to 3D. The main changes are that 4-connexity in 2D is now 6-connexity in 3D and that we deal with minimal paths and minimal action in 3D images (see section 2.1 for the 3D extension of the the fast marching).

Algorithm for Cycle detection when a region $\mathbf{R}_i$ meets a region $\mathbf{R}_j$:

```
$Test(i, j, M, i); with$
$Test(i, j, M, l);$
  • if $M(l, j) = 1$, return 1;
  • else
      – count=0;
      – for $k \in [1, N]$ with $k \neq i, k \neq j, k \neq l$ : count $+=$ $Test(k, j, M, l)$;
      – return count;
```

Table 9.2. Cycle detection

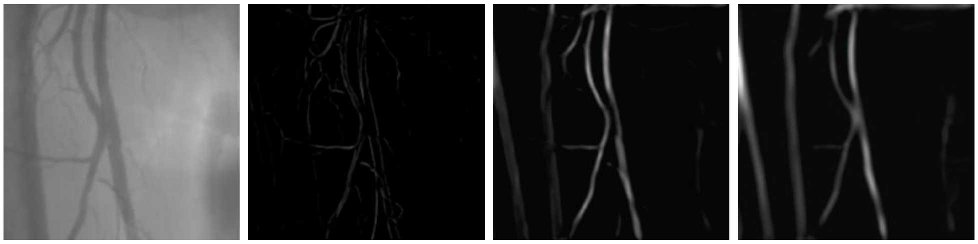


Figure 9.20. Multiscale vessel enhancement: First image is the original dataset; All other images are from left to right the filter response with respective kernels 1, 3, and 5.

Application to Real Datasets: a MR Image of the Aorta

The problem here is to complete a partially detected object. In figure 3.12 is shown a 3D MR dataset of the aorta, which presents a typical pathology: an abdominal aortic aneurysm. The anatomical object is made visible on the image by injecting a contrast product before the image acquisition.

We propose here to give a method for extracting from the grey level image a set of paths that will represent an approximate skeleton of the tree structure. This is based on extracting first a set of unstructured voxels or regions that belong to the object. Notice that [177, 107] give different methods to detect vessels but ours is much simpler and faster.

For this, we propose to extract valuable information from this dataset, computing a multi-scale vessel enhancement measure, based on the work of [60] on ridge filters. Having extracted the three eigenvalues of the Hessian matrix computed at scale σ, ordered $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$, we define the vesselness function as done in the preceding chapter 8.

In figure 9.22 you can observe the response of the filter, based on the Hessian information, at three different scales: $\sigma = 1, 5, 10$. Visualization is made with Maximum Intensity Projection (MIP). Using this information computed at several scales, we can take as potential the maximum of the response of the filter across all scales (Fig. 9.23-left). And we can easily give a very constrained threshold of this image,

Figure 9.21. ***Perceptual Grouping* on a 2D Medical Image**: left image is the vesselness potential; middle and right images show that from the set of regions obtained from thresholding of potential image, our method finds links between these regions as minimal paths with respect to the potential.

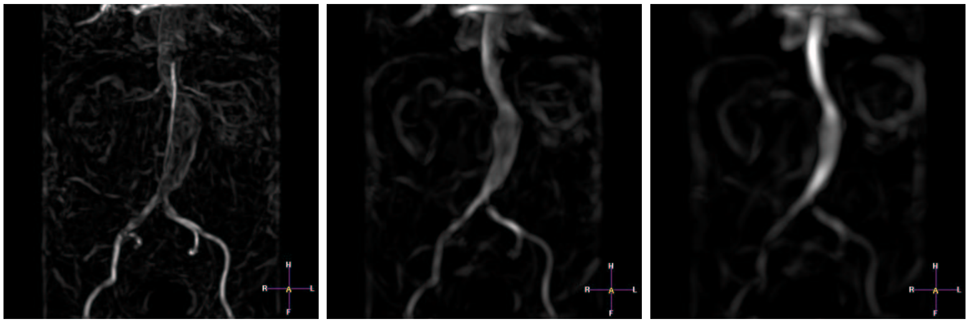


Figure 9.22. Ridge detection at three different scales: $\sigma = 1, 5, 10$ (MIP visualization of the 3D images)

that will lead to sets of unstructured voxels that surely belong to the anatomical object of interest, as shown in figure 9.23-middle.

Based on this set of regions, we apply our algorithm of section 9.3.1, using the 3D version of the Fast-Marching algorithm presented in section 2.1. We find the set of paths that connect altogether all the seed regions in our image, leading to the representation shown in figure 9.23-right.

9.3.3 Conclusion

We presented a new method that finds a set of contour curves in an image. It was applied to *Perceptual Grouping* to get complete curves from a set of edge regions with gaps. The technique is based on finding minimal paths between two end points [34]. However, in our approach, start and end points are not required as initialization. Given a unstructured set of regions, the pairs of regions that had to be linked by

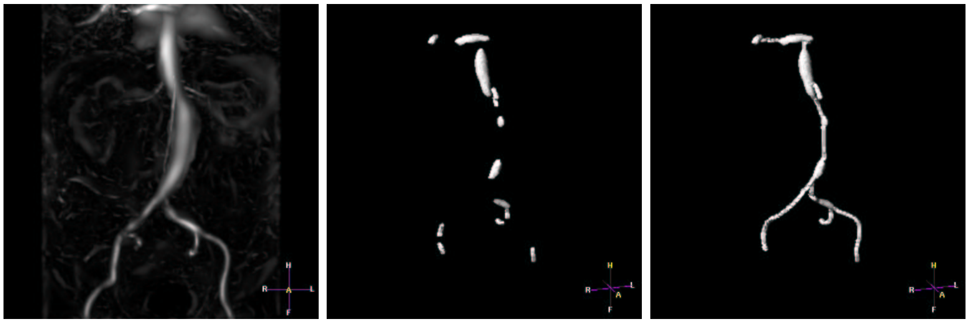


Figure 9.23. ***Perceptual Grouping*** **in the aorta of figure 3.12:** from left to right, visualization of the 3D potential (MIP view) obtained from the different scale of previous figure; a rough detection of the aorta; the Reconstructed aorta.

minimal paths are automatically found. Once *saddle points* between pairs of regions are found, paths are drawn on the image from the selected *saddle points* to both points of each pair. This gives the minimal paths between selected pairs of regions. The whole set of paths completes the initial set of contours and allows to close these contours. We applied this method in order to reconstruct vascular structures, and we showed examples for 2D vascular image and 3D medical dataset of the aorta. In case a refinement is needed, this method could be an efficient way to initialize geodesic contours. Other developments could lead to applications in roads detection in aerial images [63].